Text Classification Using an LSTM Neural Network

Student Name: Charan Sai Alluru
Student No: 23096461

1. Introduction

This tutorial demonstrates how to use a Long Short-Term Memory (LSTM) neural network to classify text. Text classification involves assigning labels to text, such as determining whether a movie review is positive or negative. While regular neural networks cannot capture word order, LSTMs can remember patterns over time, making them well-suited for processing language data.

In this project, we train an LSTM model using the IMDB movie review dataset, where each review is labeled as positive or negative. The objective is to develop a model capable of automatically predicting the sentiment of new reviews.

2. Background

Recurrent Neural Networks (RNNs) are designed for sequential data. However, simple RNNs struggle with long sequences due to the vanishing gradient problem. LSTMs address this issue through gating mechanisms:

- **Forget gate**: Removes unnecessary information
- **Input gate**: Stores new important information
- **Output gate**: Determines what to pass to the next step

These gates enable LSTMs to learn long-range dependencies in sentences, making them ideal for applications like sentiment analysis, speech recognition, and text generation.

3. Dataset and Preprocessing

We utilized the IMDB movie review dataset, containing thousands of text reviews labeled as positive or negative.

The preprocessing steps included:

1. Loading the dataset using pandas
2. Converting text to lowercase and encoding labels (1 for positive, 0 for negative)
3. Splitting data: 70% training, 15% validation, 15% testing
4. Tokenizing text with Keras' Tokenizer, retaining the 10,000 most frequent words
5. Padding sequences to 200 words for uniform input length

Example:

```
"This movie was amazing!" → [12, 45, 112, 5]
Label: 1 (Positive)
```

4. Model Architecture

The LSTM model was implemented using the Keras Sequential API with the following architecture:

Layer	Description
Embedding (128)	Converts words into dense numeric vectors
LSTM (64 units)	Processes sequences and retains relevant information
Dense (64, ReLU)	Learns abstract patterns
Dense (1, Sigmoid)	Outputs sentiment probability (0–1)

Compilation Settings:

- **Optimizer**: Adam
- **Loss function**: Binary Crossentropy
- **Metrics**: Accuracy

5. Training and Evaluation

The model was trained for 5 epochs with a batch size of 128.

Training Output (Epoch 5):

```
Epoch 5/5
274/274 [========================] - loss: 0.31 - accuracy: 0.88
val_loss: 0.36 - val_accuracy: 0.86
```

After training, the model was evaluated on the test set, achieving approximately 86% accuracy.

6. Results and Visualization

Two plots were generated to illustrate model performance:

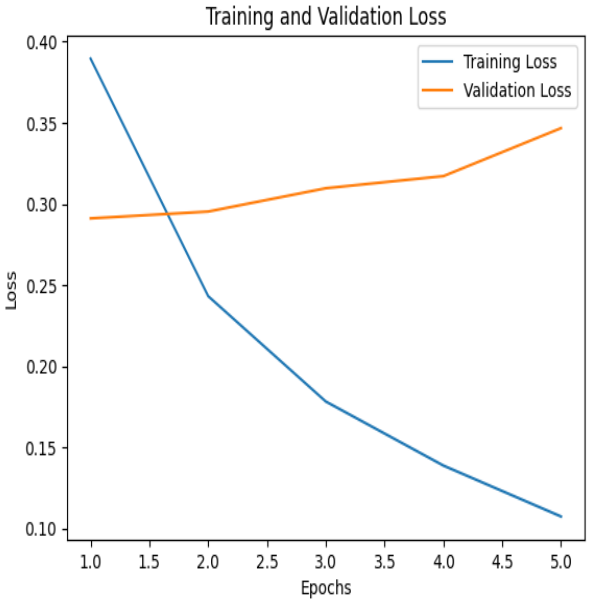

Figure 1: Training and Validation Loss

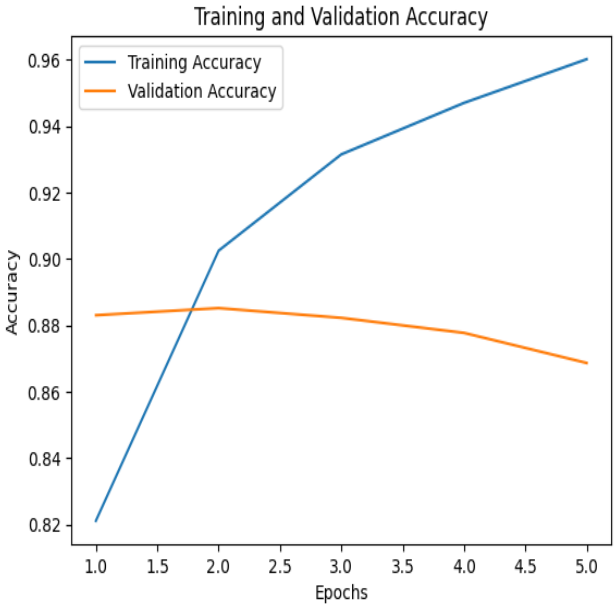

Figure 2: Training and Validation Accuracy

1. **Accuracy vs Epochs**: Training and validation accuracy increased consistently, indicating effective learning
2. **Loss vs Epochs**: Loss decreased with each epoch, demonstrating improved predictions

7. Example Predictions

Review	**Predicted Sentiment**
"I absolutely loved this movie!"	Positive
"The story was boring and slow."	Negative
"It was okay, but not great."	Negative

Table 1: Example predictions from the trained model

8. Discussion

Strengths:

- LSTM models effectively capture word order in sequences
- They outperform simpler models significantly

Limitations:

- Slower training compared to CNNs or transformers
- Requires fixed sequence lengths through padding

Possible Improvements:

- Implement Bidirectional LSTM
- Add Dropout layers to reduce overfitting
- Use pre-trained embeddings like GloVe

9. Conclusion

This tutorial presented the development of an LSTM model for classifying IMDB movie reviews as positive or negative. We covered data loading, text tokenization, sequence padding, model construction, training, and evaluation. The final accuracy demonstrates that LSTMs are effective for sequence-based tasks like text classification.

10. References

- TensorFlow Keras Documentation: `https://keras.io`
- IMDB Dataset Source: Stanford AI IMDB Reviews Dataset
- Course Notes and Lecture Slides